Chapter 15

Collaboration Paradigms and Collaborative Mechanisms

In this chapter, we offer a detailed exploration of these purposeful interactions, examining how one agent influences collaboration within MAS. We reference the diverse interaction behaviors that emerge from human social structures, further explaining multi-agent collaboration through interaction purposes, interaction forms, and the relationships that form.

Multi-Agent Systems (MAS) comprise multiple agents that interact in a shared environment, autonomously making decisions to accomplish tasks collaboratively or compete with each other [1041]. In our context, we focus on collaborative phenomenons because they widely appeared in most practical applications. Basically, each agent in MAS is equipped with different roles and initial knowledge and its own set of goals.

When engaged in problem solving or communication, agents interact with other agents or the environment to collect and process information, independently making decisions based on their objectives, existing knowledge, and observations, and subsequently executing actions [975, 1041, 1042, 1043]. Knowledge, memory, and environmental observations form the agents' beliefs, while varying motivations influence their approach to tasks and decision making [1041]. Consequently, effective problem solving requires diverse purposeful interactions, including agent-agent and agent-environment. These interactions may involve multiple rounds and occur in various directions, depending on the system design.

15.1 Agent-Agent collaboration

Considering the categorizations of MAS collaborations, we focus on more details on the granularity needed to capture the nuanced dynamics in complex multi-agent interactions. Sepecifically, we categorize inter-agent interactions into four types, inspired by sociological insights from human-to-human interaction patterns and applying them to agent-agent interactions in MAS. Sociological theories on human interaction, which include **consensus building**, **skill learning**, **teaching**, and **task division collaboration**, provide a more refined way of classifying agents. interactions. These interactions form collaborative paradigms, which enable diverse intelligent agents to work together effectively in solving complex problems, and they are shaped by various forms of goals, contexts and outcomes. Each paradigm addresses unique challenges related to cooperation,competition, coordination, and decision-making. Additionally, MAS implementations involve agents with different types of interactions, rather than a single type or unidirectional process, forming complex interaction networks that evolve over time. In collaborative software development [626, 627], a senior developer agent may interact task-wise with an architect agent, guide junior agents through multi-round dialogues. They work together on code reviews for decision-making and learn with a testing expert agent to improve test coverage. Examining the objectives and results of these interactions reveals the crucial techniques and technologies shaping agent behavior and decision-making, thereby enhancing our comprehension of multi-agent dynamics.

Consensus-oriented Interaction Consensus-oriented interactions concentrate on harmonizing the MAS's final target via negotiation, voting, and social choice frameworks [1044]. This interaction is significant for incorporating diverse knowledge and ensuring agents shift their views towards a unified understanding to achieve consensus [1045]. In this interaction, agents integrate knowledge to establish a unified understanding, which largely helps joint decision-

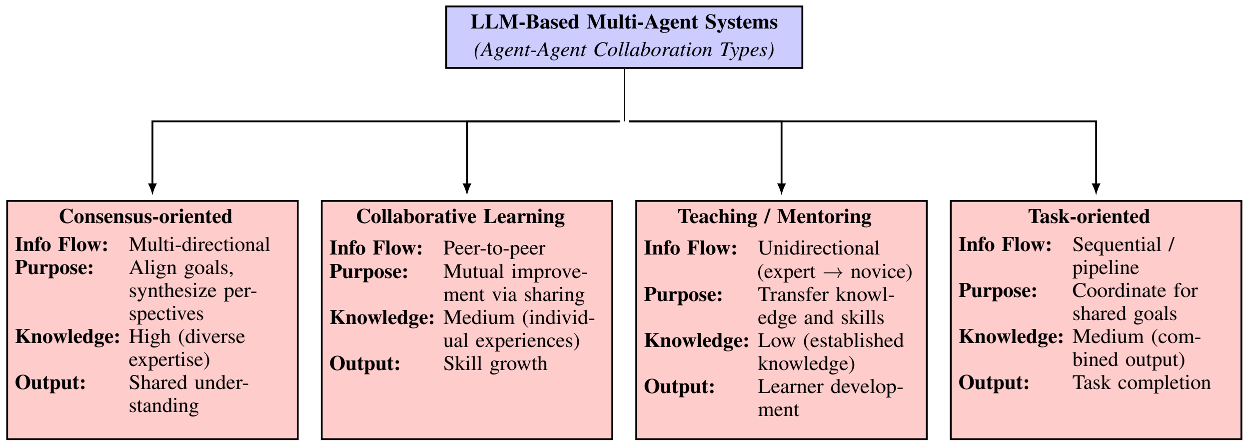

Figure 15.1: An overview of four agent-agent collaboration types in LLM-based MAS: *Consensus-oriented*, *Collaborative Learning*, *Teaching/Mentoring*, and *Task-oriented*. Each type is described along four key dimensions: information flow, collaboration purpose, knowledge integration, and output focus.

making in complex problem-solving situations that demand different viewpoints. For instance, MedAgents [922], MDAgents [1046], and AI Hospital [1036] demonstrate how collaborative dialogue among multidisciplinary agents improves problem solving by sharpening reasoning skills and accessing inherent knowledge.

These dialogues allow agents to ensemble expertise into coherent outcomes, frequently outperforming conventional methods like zero-shot or few-shot reasoning. The importance of consensus-driven teamwork is particularly evident in scientific environments, where addressing complex challenges requires diverse perspectives and meticulous validation. Agent Laboratory [746], serves as an example where PhD and postdoctoral agents collaborate to agree on research objectives, interpret experiments, and consolidate research findings. Similarly, Virutal Lab [752] organize a series of team to conducts scientific research, where all agents discuss a scientific agenda, and individual meetings, where an agent accomplishes a specific task.

Methods for multi-agent consensus typically include several approaches, including **Discussing**, **debating**, **negotiating**, **reflecting**, and **voting**. Common methods for reaching consensus encompass an array of structured techniques. The primary mechanisms involved are **discussing**, **debating**, **negotiating**, **reflecting**, and **voting**. Debates allow agents to obtain competing hypotheses, while negotiation helps resolve conflicting priorities and resource limitations. Specific frameworks have been created to support these consensus-building activities. During these processes, agents gather outputs from peers tackling the same issue, and include environmental feedback as numerical data and contextual details. These interactions enable agents to share viewpoints, assumptions, and progressively achieve a common understanding.

For example, GPTSwarm [651] formulates the collaboration between agents with graph design, that the information flow and edge connections build the basic group discussion. In GPTSwarm, if an agent consistently provides incorrect opinions, it will be excluded. RECONCILE [918] uses a round-table discussion format with several discussion cycles and voting systems based on confidence levels. It integrates reflection by learning from past discussions, using confidence metrics and human insights to improve their responses. Furthermore, debates are quite important for achieving agreement, reducing hallucinations and also addressing complex issues [985, 1047, 1031, 1003]. In GOVSIM [1048], agents collaborate to achieve a balance, and it suggests using a shared resource and conserving it for future needs. The negotiations went beyond simple information exchange and relationship-focused interactions. The Multi-Agent Debate (MAD) framework [1031] promotes creative thinking by having agents deliver arguments in a "tit-for-tat" pattern, with a judge overseeing the process to finalize a solution. The Formal Debate framework (FORD) [1004] enhances consistency among language models through organized debates, enabling stronger models to steer consensus, while weaker ones adjust their perspectives. Similarly, AutoAgents [1030] define a collaborative refinement action in which each agent updates its chat record. In the process, it also appends the previous statements of the other agent and refines its action to achieve consensus.

Collaborative Learning Interaction In collaborative learning, interaction usually happens among similar agents. Although architecturally alike, accumulate distinct memories and experiences due to their unique behaviors and varied environmental interactions. By solving problems together, these agents share experiences to boost their strategy learning, task-solving, and skill acquisition capabilities. Over time, each agent enhances its skills through ongoing interaction, leading to the evolution of individuals. The key difference between collaborative learning and consensus-oriented interactions lies in their fundamental goals and processes. While consensus-oriented interaction focuses on knowledge integration and belief alignment through synthesizing diverse viewpoints to reach agreement, collaborative learning interaction emphasizes peer knowledge construction and experience sharing, prioritizing mutual improvement and individual growth. When engaged in collaborative learning interaction, agents update their context or memory from observing others' behavior. For example, agents can learn optimal strategies by observing the deliveration from peers, adapting their own approach based on these observations without necessarily agreeing on a single "best" strategy [961, 962, 963, 971, 965, 967, 972, 968, 969]. As highlighted in [966], the effective discussion tactics significantly impact learning outcomes among agents. In these interactions, agents collaborate to learn and address problems, focusing on mutual understanding and enhancement rather than reaching unanimous decisions. This method refines personal responses and knowledge via ongoing feedback.

The methods commonly employed in collaborative learning interaction include: **1). Experience sharing.**, Agents exchange personal insights and best practices. As described in [303], iterative experience refinement enables LLM agents to achieve adaptive improvement in software development via continual acquisition and utilization of team experience in successive pattern and the cumulative pattern. Furthermore, MAS-CTC [301] is a scalable multi-team framework that enables orchestrated teams to jointly propose various decisions and communicate with their insights in a cross-team collaboration environment. It enables different teams to concurrently propose various task-oriented decisions as insights, and then communicate for insights interchange in important phases (multi-team aggregation). Different agent teams utilize a greedy pruning mechanism and aggregation mechanisms to eliminate low-quality content, thus improve the performance in software development. Differently, in MOBA [1049], a novel MLLM-based mobile multi-agent system, global agent reflects on local agent execution results to support adaptive planning to align with the environment. AutoAgents [1030] employs a knowledge sharing mechanism where agents exchange execution results to enhance communication and feedback, where agents can obtain long-term, short-term and dynamic memory from others. **2). Peer discussions.** Peer discussions allow agents to articulate their reasoning processes and learn from others' approaches. MEDCO [923] create a dynamic environment where clinical reasoning and decision-making skills are strengthened through collaborative problem-solving among student agents. Moreover, In [1050], agents engage in structured peer discussions after initializing their output, reviewing each other's reasoning step by step. Through feedback exchange and confidence scoring, agents refine their decision-making, learn from diverse approaches, and iteratively enhance their reasoning accuracy, fostering collaborative knowledge acquisition. **3). Observational learning.** Observational learning occurs when agents monitor others' behaviors and outcomes to inform their own strategies. AgentCourt [1051] develops lawyer agents that participate in court debates and improve through accumulated experiences, demonstrating improved reasoning and consistency through experiential learning. In iAgents [1046], the human social network is mirrored in the agent network, where agents proactively exchange human information necessary for task resolution, thereby overcoming information asymmetry. iAgents employs a novel agent reasoning mechanism, InfoNav, to navigate agents' communication towards effective information exchange. Together with InfoNav, iAgents organizes human information in a mixed memory to provide agents with accurate and comprehensive information for exchange. Additional experimental phenomenon indicates difficulty of certain tasks making agents continuously refine their strategies in pursuit of the required information. MARBLE [948] designs a cognitive evolve planning combining the 'expectation' of the agent and its actual action results to update the overall planning experience for better planning in the next round.

Despite its benefits, collaborative learning interaction faces several challenges. These include ensuring equitable knowledge exchange among agents with varying capabilities, preventing the propagation of errors or biases across the system, maintaining agent diversity while facilitating learning, and developing effective mechanisms for agents to selectively incorporate others' knowledge based on relevance and reliability. Overcoming these challenges requires the meticulous creation of interaction frameworks and learning strategies. And it should balance individual advancement with the broader development of the system. Although issues such as knowledge fairness, bias propagation, and scalability present difficulties, there is great potential to improve MAS, particularly in dynamic and complex environments. By using iterative learning processes and providing opportunities, collaborative learning enables agents to develop richer knowledge bases and more refined problem-solving abilities.

Teaching/Mentoring Interaction To tackle these challenges, it is important to carefully develop interaction protocols and learning frameworks that harmonize individual development with overall system progress. In the context of MAS, teaching and mentoring interactions are fundamental mechanisms in collaborative environments, especially in scenarios where knowledge transfer is essential for growth and collective intelligence. Unlike collaborative learning, where knowledge is exchanged reciprocally among agents, teaching and mentoring interactions focus on the unidirectional flow of knowledge from an experienced agent to a less experienced one. The mechanisms and methods used in teaching/mentoring interactions include several key strategies:

- **Criticism and Feedback.** The mentor agent evaluates the learner's performance and provides corrective or constructive feedback. This helps the learner refine their knowledge and skills through a feedback loop where they update their internal knowledge based on the feedback received.
- **Evaluation.** Mentors assess the learner's capabilities or progress through performance reviews and clear assessment criteria, providing valuable insights for development.
- **Instruction and Teaching.** Mentors convey targeted knowledge, guidelines, or techniques using direct instruction which allow learners to pose questions and receive clarifications.

Iterative Teaching and Reinforcement Teaching is typically progressive, where each phase provides opportunities for the learner to complete tasks and get feedback. For example, in the MEDCO system [923], student agents improve their professional skills through a cyclic practice-oriented learning approach directed by expert mentors, in addition to engaging in peer discussions. These expert agents conduct ongoing assessments and provide real-time guidance on clinical competencies, focusing on patient interaction skills and diagnostic reasoning. [921] shows that an agentic doctor can continually improve their diagnosis by merely interacting with agentic patients in a simulated hospital and can transfer its learned knowledge of real-world cases.

This interaction type can be categorized based on the direction of knowledge transfer into two primary types: unidirectional and interactive. Unidirectional is rooted in traditional teaching models where knowledge flows from the teacher to the student. This approach emphasizes the transmission of facts and concepts, often involving lectures and direct instructions [923].

Task-oriented Interaction. Task-oriented collaborations involve agents working together to achieve common objectives through effective coordination and task decomposition strategies, as well as a high degree of cooperation and coordination. Agents interact primarily by processing upstream output and generating results for downstream agents following established task dependencies rather than engaging in complex discussions or debates.

Recent frameworks demonstrate diverse implementations of this interaction pattern: **(1) software development frameworks** such as MetaGPT [626] and ChatDev [627], agents operate in a structured pipeline that mirrors the software development lifecycle. For example, architect agents process requirements to generate technical specifications, which development agents then use to produce code, followed by testing agents who validate the implementations; **(2) Collaborative reasoning** frameworks like Exchange-of-Thought (EoT) [1052], GPTSwarm [651], MACNET [1028] involve structuring agents in a specific format (e.g., ring, tree, directed acrylic graphs, optimizable graphs) , which mitigates context expansion risks by ensuring only optimized solutions progress through the sequence, and enforcing multiple agents to collaborate together towards solving complex mathematical or knowledge reasoning tasks; In **(3) ML applications** [1053, 1019], agents adhere to stringent workflow structures, each fulfilling specific tasks in processes. For more complex tasks such as VideoQA, the TraveLER framework [1054] showcases modular task breakdown across structured phases (Traverse, Locate, Evaluate, and Replan), with a Planner agent managing interactions and improving strategies based on iterative agent inputs.

These handoffs rely on explicit deliverables instead of direct agent negotiations. Inspired by GPTSwarm [651]-alike graph agentic systems, MACNET [1028] structures agents into directed acyclic graphs (DAG). Here, supervisory figures issue directives while executors implement solutions. By ensuring only optimized solutions progress through the sequence, this setup mitigates context expansion risks. In ML applications [1053, 1019], agents adhere to stringent workflow structures, each fulfilling specific tasks in processes. For more complex tasks such as VideoQA, the TraveLER framework [1054] showcases modular task breakdown across structured phases (Traverse, Locate, Evaluate, and Replan), with a Planner agent managing interactions and improving strategies based on iterative agent inputs.

Beyond organized development, task-driven interactions have been shown in open-ended contexts such as Minecraft game, in where agents adjust to ever-changing environments. In [927], leader agents manage workflows by breaking down complex objectives into specific tasks, while executor agents perform actions like gathering resources. Coordination mechanisms are important for ensuring agents collaborate effectively towards final goal, including communication protocols, synchronization strategies, and resource-sharing techniques. The interaction of agents in MAS for task execution has garnered significant interest, notably through utilizing LLMs for handling intricate tasks and workflows. The collaboration of agents are vital for task completion, particularly in ever-changing settings like software development and project management [626, 630].

15.2 Human-AI Collaboration

To unlock the potential of MAS in meeting human objectives, people often work alongside them using three primary methods: **one-off task delegation**, **multi-turn interactive instruction**, and **immersive human-agent collaboration**.

In **one-off task delegation**, humans delegate single-instance tasks to MAS, such as posing a question to a Q&A platform or assigning a coding task [1055, 626]. Without additional input, the agent handles the task autonomously, delivering a complete response or solution in a single reply. This is presently the prevalent way humans collaborate with LLM-based agents [922, 627, 31].

For **multi-turn interactive instruction**, humans engage in iterative interactions with LLM-based agent systems to refine and explore solutions until a satisfactory result is achieved. This type of interaction is widely seen in creative applications, such as image editing or writing edit [938]. For instance, a user might ask the system to add an object to a specific location in an image, replace an element, change the background, or revise a part in a sentence. These interactions often span multiple rounds, with users continuously refining their requests until the desired outcome is reached. Moreover, certain other LLM-based agent systems may require human approval or clarification during multi-turn interactions before proceeding to the next step [1056, 930]. Under human guidance, these LLM-based agent systems can complete household tasks as well as software development tasks.

Immersive human-agent collaboration features LLM-based agents simulating human behaviors to serve as partners. For instance, in an immersive setting, humans treat these agents as teammates, achieving common objectives. Instances include agents representing humans in meetings or help solve tasks like chores or projects. This strategy highlights effective integration and teamwork in dynamic contexts [937, 924].

To assess Human-AI collaboration quantitatively, several frameworks have been suggested. Co-Gym [1057], for instance, measures the communication, situational awareness, and personalization of LLM-based agents in tasks such as travel planning, writing related work, and tabular analysis.

In summary, as LLM-based agent systems have advanced, Human-AI collaboration has diversified to address challenges across domains. This ranges from simple command-based AI interactions for questions, to multi-turn dialogues for design and development, and partnering with human daily tasks.

With advancements in LLM-based agent systems, they are expected to integrate more into daily life, streamlining tasks and boosting efficiency. At the same time, humans will refine and adapt their ways of interacting with AI, leading to more effective collaboration. We believe this shift will drive fundamental changes in both social productivity and the social relations of production, reshaping how work is organized and how humans and AI cooperate in the large language models era.

15.3 Collaborative Decision-Making

Collaborative decision-making processes are crucial for ensuring the efficient operation of MAS and the successful completion of tasks. Although collaboration itself is a core feature, the approaches of decision-making directly determines the effectiveness of collaboration and the overall performance of the system. Recent research has highlighted the critical role of collaborative decision-making. [1037] showed that diverse decision-making methods can significantly enhance the collaborative efficiency of the system. [649] emphasized that a rational decision-making mechanism can stimulate the emergence of intelligence within a system.

From a broader perspective, the collaborative decision-making process can be divided into two major categories based on their architectural characteristics: Dictatorial Decision-Making and Collective Decision-Making [1037].

Dictatorial Decision-Making. Dictatorial Decision-Making is a process where decision-making relies on a single agent in a MAS. In this paradigm, all agents send their state information or local observations to this dictatorial agent. The dictatorial agent is responsible for assembling this data, studying the core problems, and establishing definitive decision guidelines. The key principle for such an approach is to leverage a global mindset in moving towards improved decision-making, hence paving the reliability of the system performance along with the successful achievement of task goals. [1031, 1058, 1046] demonstrated the single-agent decision-making process with a single LLM, who synthesized various views on the same problem to make decision-making even more objective and comprehensive. Furthermore, [134, 1059] suggested the weighted integration method through ranking, scoring or checklist, enhancing the robustness of decision-making procedures. In addition, beyond the explicit inclusion of perspectives, [1030, 1060] proposed architectures where a central agent breaks down complex tasks into simpler sub-tasks and assigns them to specialized agents grouped by their functionalities. Moreover, in [651, 1028],it is common that the last node's agent works in an environment to assemble the past information and deduce a conclusion according to the topological structure, rather than by a central agent.

Collective Decision-Making. Collective Decision-Making involves agents collaborating to reach decisions without a central authority, relying on local data and interactions like voting or negotiation. This method shares decision-making power among agents, allowing the system to adapt according to changes while maintaining robustness and scalability.

- ***Voting-based Decision Making***Voting systems are important for collective decision-making, providing a framework for reaching consensus. A conclusive majority is achieved through voting as described by [1045, 968]. Moreover, the GEDI electoral module [1037] enables multiple voting methods. This method largely improve reasoning and fault-tolerance while avoiding complex system designs.
- ***Debate-based Decision Making***In comparison with voting-based methods, debate-based decision-making focuses on organized interactions between agents, in order to obtain the best result. In [1031, 1061], agents participate in guided discussion, where they articulate and proposals in an attempt to resolve disagreements and reconcile points of view. Simultaneously, [1050, 1062] practice restraint stance, using communication channels among agents for consensus-building through repeated discussions. To tackle the issue of "cognitive islands," certain systems would employ a common retrieval knowledge base to enable agents to be aware of the same knowledge throughout debates [1005]. By mimicking human dialogue, these systems allowed agents to exchange perspectives and make more informed decisions.

Discussion and Future Work Collaboration in multi-agent systems (MAS) still faces numerous challenges that require further research. Current methods are largely based on contextually dependent interactions; however, they do not include a specific framework for training and optimizing cooperative actions. This heavy dependence on large language models (LLMs) has some limitations, as their effectiveness is inherently tied to the size of the LLM's contextual window and its native reasoning capabilities. While LLMs provide a solid foundation for enabling interactions, these systems are still limited by the inherent limitations of context-dependent communication.

Future studies should focus on finding frameworks that inspire agents for active learning with regard to optimal timing and information dissemination methodologies. Using methodologies from multi-agent reinforcement learning (MARL), there is a growing requirement for strategies that will help agents determine appropriate moments for information sharing, as well as what information should be shared through what channels. This calls for not just devising novel interaction protocols but also incorporating training methodologies that will constantly optimize these protocols with each improvement.